Project Final Report
ECE 1786

Song Genre Classifier

Andrea Haw (1009238969)
Taikun Zhang (1004050643)

Word count: 1896 words
Penalty: 0%

December 13, 2022

1. Introduction

Music is an essential part of society and culture. Popular music streaming platforms such as Spotify and Apple Music use song metadata to classify songs by their genres . The purpose of our project is to create a classifier that will predict the music genre of a song given an input sequence of its lyrics. The 8 genres that we chose for our classifier were: R&B, Pop, Rock, Hip Hop, Blues, Country, Indie, EDM. Our application would be useful in recommendation algorithms for suggesting new music based on a user's genre preferences.

2. Illustration/Figure

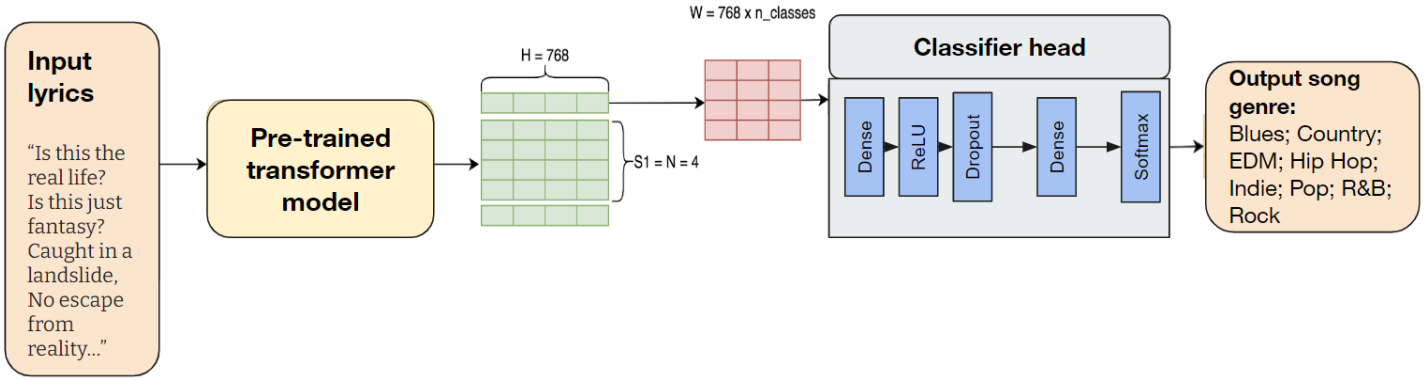


Figure 1. Illustration of Model Architecture

3. Background and Related Work

While the music industry has focused on song genre classification using a combination of lyrics, audio signals, and musical data [1], the efforts to predict song genre solely based on lyrics are limited.

A project from Stanford attempted to create a classifier using a LSTM model and GloVe embeddings [2]. Their dataset consisted of 6 different genres with songs in multiple languages. During training, each song was given multiple genre labels to help with correct predictions. The LSTM model was able to classify genres even with an unbalanced dataset but had difficulties when trying to distinguish longer songs [2].

Another project used data from Million Song Dataset (MSD) to implement classification of seven genres with Bag of Words and Part of Speech (POS) features [3]. After generating the most common words and words tagged in POS, the results show that there are lyrical differences between genres but also similarities [3]. A 65.71% accuracy was reported for the trained Naive Bayes model [3].

4. Data and Data Processing

For our dataset, we took a combination of two datasets from Kaggle [4,5]. We filtered out unrelated genres and songs that were not in english. After analyzing the number of data samples per class, we scraped Spotify's web API for more songs for genres that were underrepresented. The number of data points per genre is shown below.

```
df['primary_genre'].value_counts()

Rock       25177
Pop        13759
Indie      12998
Hip Hop     8412
Country     7377
R&B         5309
Blues       2038
EDM         1758
```

Figure 2. Number of data points per primary genre

Other things we included into our dataset were multiple genre labels for each song. Each song has a field of a "primary" and "secondary" genre. This feature of the data is something we do when fine-tuning our model because songs can fall under multiplier genres and this would overall help the training of our model. Our data preprocessing involved tokenizing, padding, removing stop words, and encoding the genres for our preliminary models. Here is an example:

```
track_name                                          Careless Whisper
lyrics             I feel so unsure\nAs I take your hand and lead...
primary_genre                                                    Pop
secondary_genre                                                  NaN

 genres                                                        Pop
 lyrics    I feel unsure As I take hand lead dance floor ...
```

Figure 3. Before and after of an example data point that is processed

When looking deeper into the data, some genres have more unique words than others.

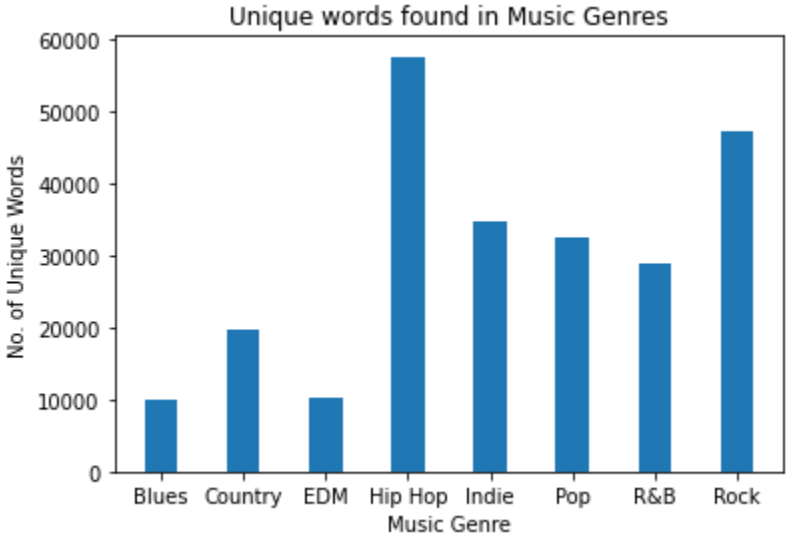


Figure 4. Number of unique words per genre

There are also many common unique words per genre. We noticed that Rock, Indie, Pop and R&B were very similar. Similarities between the lyrics of these genres were measured to be greater than 50%. Below are some word clouds of the most common words in these genres.

Figure 5. Word Cloud for Pop

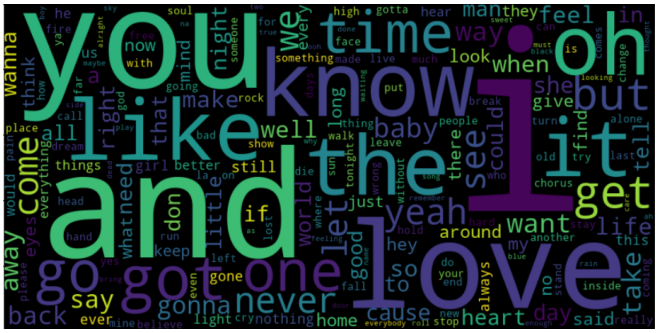


Figure 6. Word Cloud for Rock

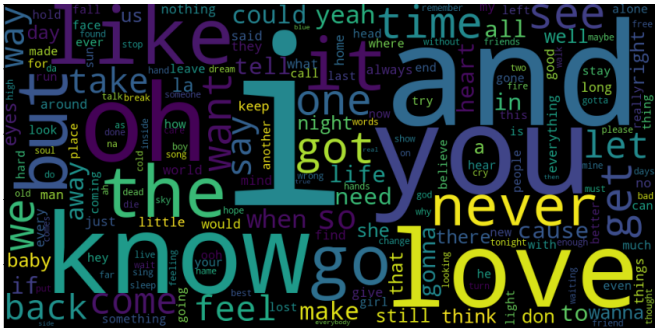


Figure 7. Word Cloud for Indie

Figure 8. Word Cloud for R&B

5. Architecture and Software

We decided to focus our efforts on using state-of-the-art transformer models over LSTMs and RNNs like previous work. We fine-tuned a distilled version of GPT-2 (DistilGPT2) with 2 linear layers as the classification head. The base DistilGPT2 model has 6 layers, an embedding dimension of 768, and 12 attention heads, with masked self-attention being applied in the decoder blocks [6]. There are around 82 million parameters in total. To reproduce a model similar to ours, one must first load the pre-trained "distilgpt2" model using the Huggingface GPT2Tokenizer and GPT2 Model classes. An additional classifier head shown in figure was also added to the model. Dropout was added with the intention of reducing overfitting.

```
self.classifier_head = nn.Sequential(
                            nn.Linear(embedding_size*max_seq_len, 768),
                            nn.ReLU(),
                            nn.Dropout(0.1),
                            nn.Linear(768, num_classes)
                           )
```

Figure 9. Pytorch code for Classifier Head

A custom pytorch Dataset class was created for the specific lyric-genre dataset, and for the initial model a training loop was created, using cross entropy as the loss function. To support training with multiple labels, the cross entropy loss was calculated for each label (primary and secondary if available) and the larger one was used for backpropagation. For the accuracy calculations, the model prediction was accepted if it either matched the primary or secondary label.

Other software the team created included code to explore and analyze the dataset, and a basic Gradio interface that takes an input of lyrics and outputs the predicted genre.

6. Baseline Model

For our baseline model, we decided to use a term frequency-inverse document frequency (TF-IDF) model which statistically finds the relevance of each word in a given text. We used the TF-IDF functions from sklearn and calculated frequencies as inputs for four simple classifier models: Random Forests, MultinomialNB, LinearSVC, and LogisticRegression. LogisticRegression gave the highest accuracy of 40.6 % and there was a lot of confusion in classifying songs between Indie, Pop, and Rock. To give an idea of what the model uses to distinguish between classes, Figures 10 and 11 show the feature importances for the top weighted TF-IDF tokens.

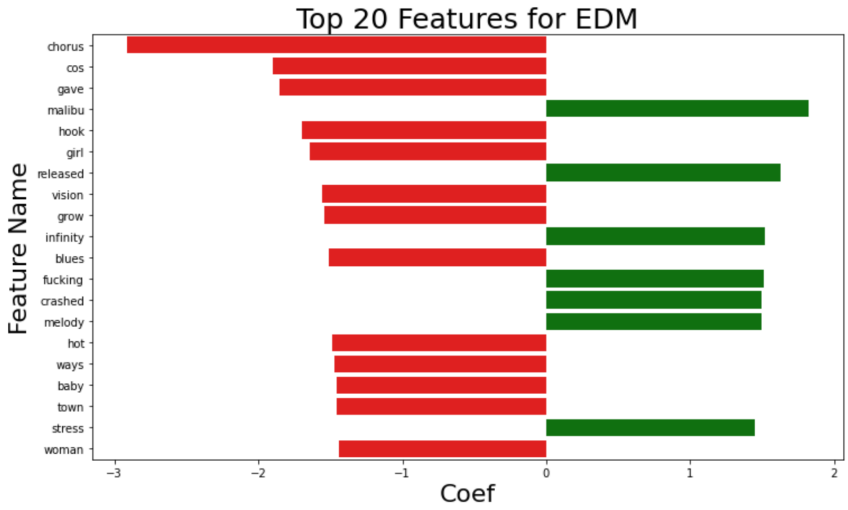


Figure 10. Feature importances of the baseline model for the EDM genre

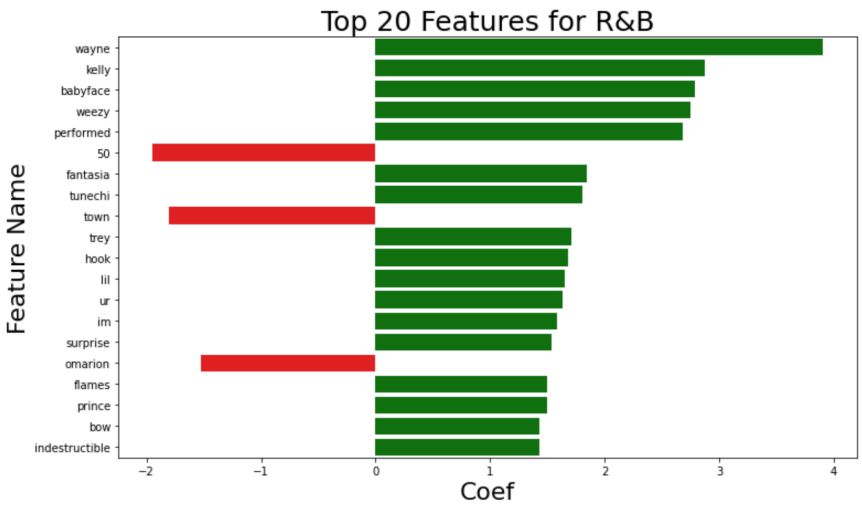


Figure 11. Feature importances of the baseline model for the R&B genre

7. Quantitative Results

For our baseline model, the highest accuracy achieved was 40.7% using Logistic Regression. Looking at the accuracy per class and confusion matrix, it seemed like the baseline model had a very hard time distinguishing between Pop, Rock, and R&B.

	precision
Blues	0.53
Country	0.42
EDM	0.44
Hip Hop	0.66
Indie	0.30
Pop	0.30
R&B	0.37
Rock	0.22

Figure 12. Precision and Confusion matrix for baseline model

The confusion matrix in Figure 13 shows the results of the DistilGPT2 model trained with default hyperparameters and using only the primary genres as labels. Similar to the baseline, the model performed quite poorly on the pop, R&B, Indie and rock genres. There is a fair amount of confusion between these genres as shown by the higher percentage of incorrect labels among these categories.

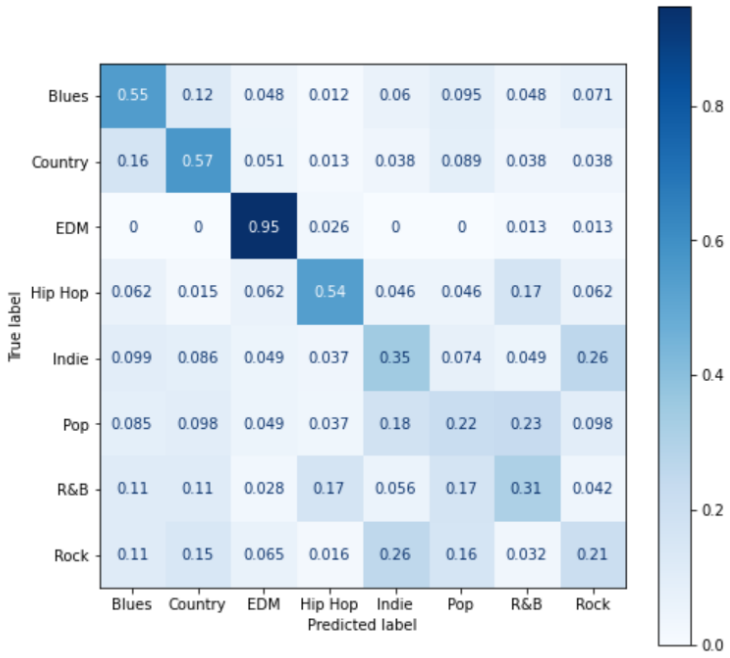


Figure 13: Confusion matrix for model trained on primary labels

The best performing model achieved was the DistilGPT2 version with tuned hyperparameters with a test accuracy of 61.6%. The confusion matrix for the holdout test set is shown below, with notable improvements in the Hip Hop and R&B categories.

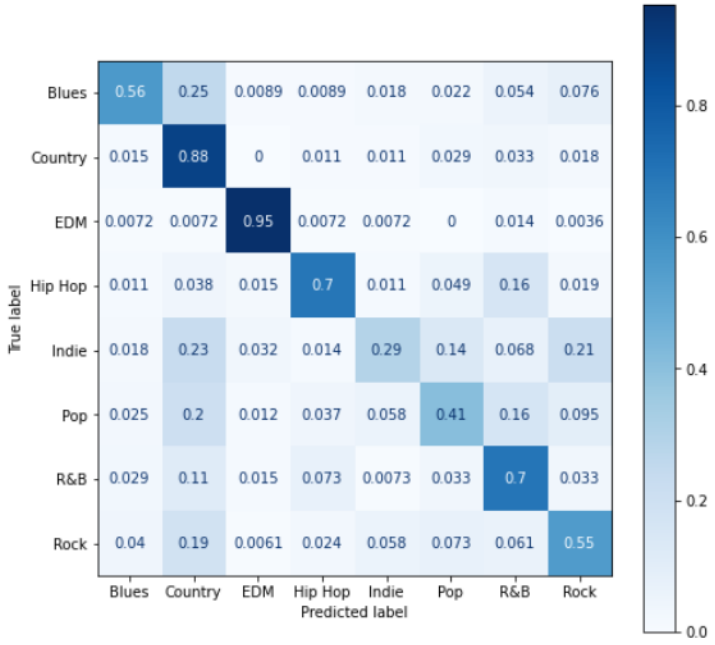
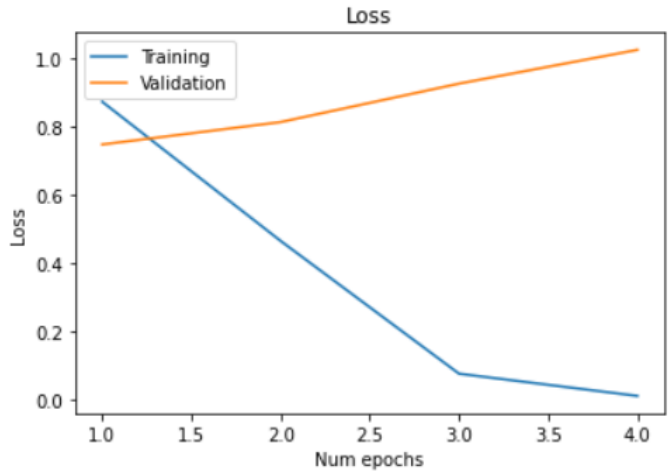


Figure 14: Confusion matrix for final model trained with primary and secondary genre labels7



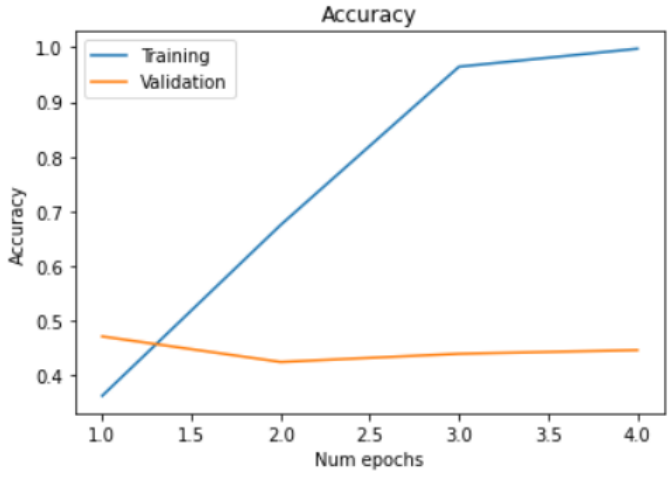
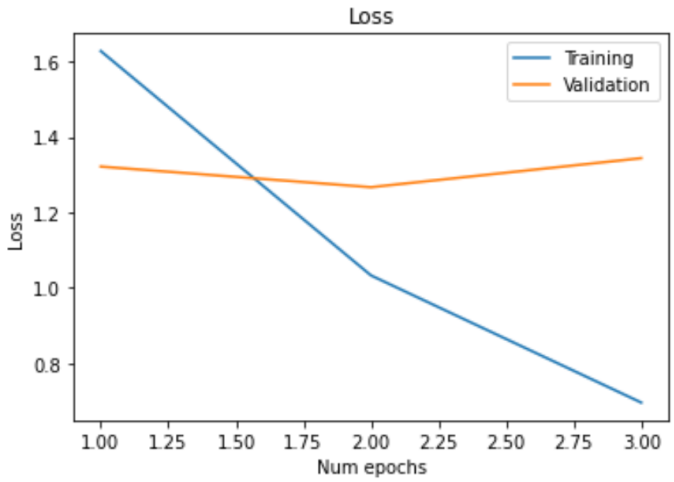


Figure 15a. Training and validation curves for the model trained on primary labels



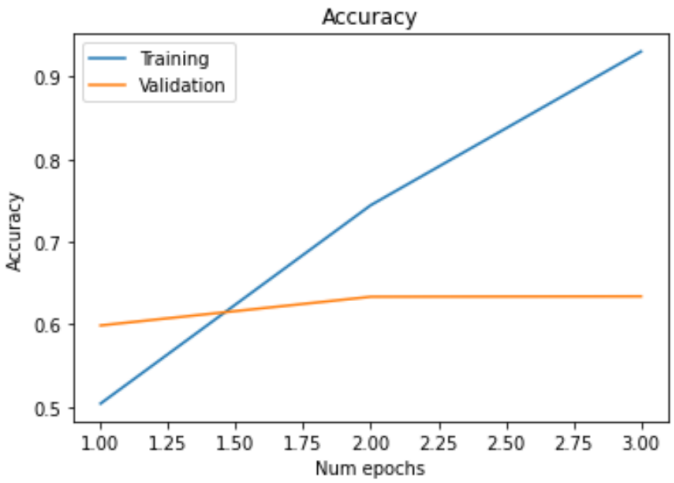


Figure 15b. Training and validation curves for the final model

8. Qualitative Results

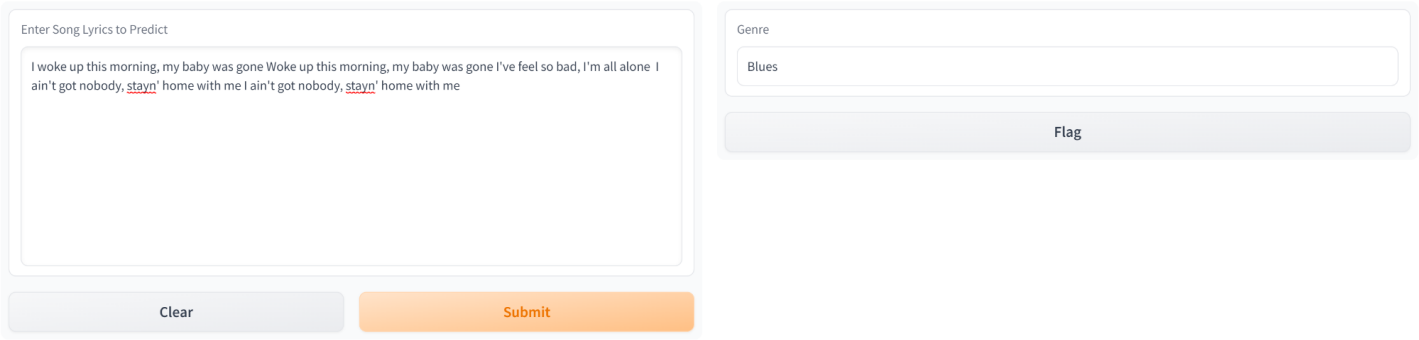


Figure 16. Example of correct Blues prediction

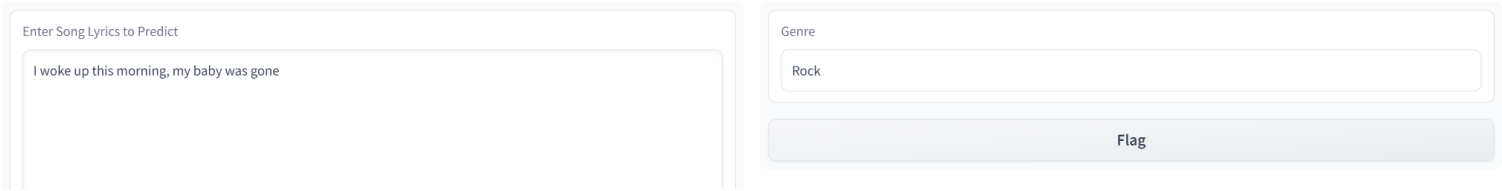


Figure 17. Example of incorrect prediction of Blues genre

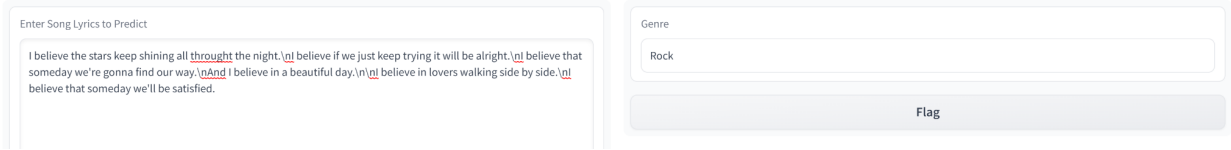


Figure 18. Example of correct Rock song prediction

EDM:

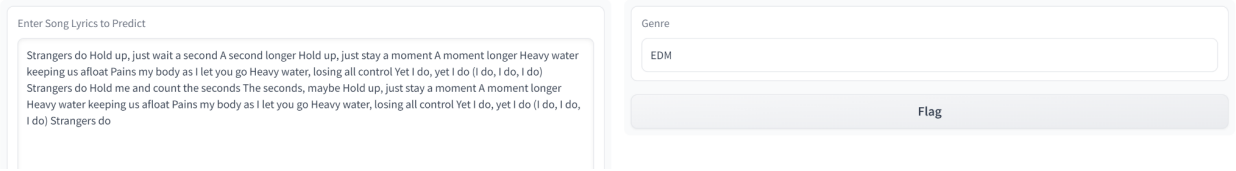


Figure 19. Example of correct EDM song prediction

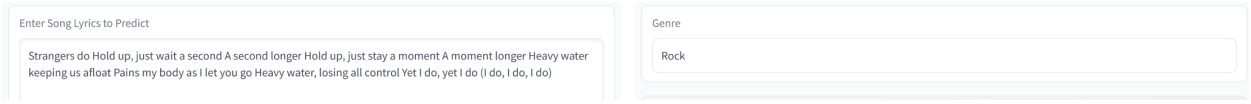


Figure 20. Example of incorrect prediction of EDM genre

9. Discussion and Learnings

The initial training and validation curves in Figure 15a clearly show the model being overfit in 4 epochs, which may seem odd if the neural network was trained from scratch, but is not surprising considering this was a fine-tuning classification task. The validation accuracy is notably poor as compared to the training accuracy, suggesting that the model is not generalizing on new data. Regarding the final model's loss and accuracy curves (Figure 15b), one could argue that the model is still overfitting; however, the validation loss has not continually increased after each epoch, and the accuracy increased by around 20 percentage points, from 43% to 63%. While 63% does not seem to be a convincingly successful number for accuracy, it is actually quite commendable given the eight possible classes for the model to predict. The Stanford project reported a 68% accuracy [2] for only three of the eight classes that were used in this project.

One of the main issues we addressed was the multi-label problem due to genre overlap. By modifying the loss function and enabling multiple label prediction outputs for accuracy scores, we were able to find a lift in performance. The genres not being mutually exclusive still poses a problem for the final model. This is evident with the existence of many secondary genres for songs and similar genres often being paired together as primary/secondary genres. We experimented with training the model without the Indie and Pop genres in an attempt to improve performance in distinguishing between songs, since Indie is considered a subgenre of Rock or Pop, and Pop is representative of popular music, which may encompass different genres. Performance did improve as expected when removing these two genres, but we kept all genres in for comparison to address the original task at hand.

Based on the qualitative results, we see that the model is effective in identifying the patterns or structure unique to different genres, as shown by its correct Blues, EDM and Rock predictions in Figures 16-20. For example, the model was able to identify the repetition and words often used in Blues and EDM songs. When we truncated the same input lyrics of the Blues song, the model predicted Rock instead. The same observation holds for truncating the lyrics of an EDM song and removing the element of repetition.

Table 1. Class distribution of common primary and secondary genres

Primary Genre	**Secondary Genre**	**# of songs**
Indie	Rock	4518
Pop	R&B	2174
Rock	Indie	1817
Hip Hop	R&B	1802
Rock	Blues	1493
R&B	Pop	948

From the distribution of classes shown in Table 1, we note that Hip Hop and R&B, and Rock and Blues are two pairs of genres that songs are often labeled together as, which makes sense considering the history behind the musical styles.

Overall, we learned that the model was successful, but limited in its task of predicting genres - to be accurate and more confident about these predictions, it would be best to use them alongside other audio features to give stronger signals. We noticed how prevalent the risk of overfitting is, even for 3-4 epochs when fine-tuning large transformer models for a specific task.

If we were to approach the project again, we would do a more thorough exploration of the data, especially studying the types of words used in each genre for song lyrics so we could have more time to solve the issue of overlapping classes. Also, we would refine our dataset to add songs that are less controversial between 2 genres. Finally, adding extra features related to the song lyrics such as length, verses vs chorus, and artist would help enrich our dataset and further reduce overfitting.

10. Individual Contributions

Table 2. Individual contribution tasks

Task	**Person Responsible**
Dataset curation from kaggle and queries	Both
Querying Spotify API for songs and their genres	Taikun
Querying Musixmatch API for song lyrics	Andrea
Manual labeling and combining datasets	Taikun
Processing and exploration of the classes/labels	Andrea
Baseline model training	Taikun
GPT2 model training	Andrea
Interpreting and evaluating initial results	Both
Bert model training	Taikun
Multiple Label Training and Prediction	Andrea
Analyzing Important Features of Baseline model	Andrea
Analyzing Words in the Dataset	Taikun
Wrote Gradio code for user side	Taikun

We split up the tasks and worked in parallel to complete the project. The tasks were delegated with divided efforts focusing on dataset, text, and label preparation in addition to training the baseline and transformer models. The table above gives a more in depth division of the tasks and responsibilities.

Permissions

Team Member	Post Video?	Post Final Report?	Post Source Code?
Andrea Haw	Yes	Yes	Yes
Taikun Zhang	Yes	Yes	Yes

References

[1] S. Oramas, O. Nieto, F. Barbieri, and X. Serra, "Multi-label Music Genre Classification from Audio, Text, and Images Using Deep Features," *ISMIR*, Jul. 2017.

[2] A. Boonyanit and A. Dahl, "Music Genre Classification using Song Lyrics," 2021.

[3] J. Yang, "Lyric-Based Music Genre Classification," 2014.

[4] A. Neisse, "Song lyrics from 79 musical genres," *Kaggle*, 17-Mar-2022. [Online]. Available: https://www.kaggle.com/datasets/neisse/scrapped-lyrics-from-6-genres. [Accessed: 16-Nov-2022].

[5] M. Nakhaee, "Audio features and lyrics of Spotify Songs," *Kaggle*, 14-Jun-2020. [Online]. Available: https://www.kaggle.com/datasets/imuhammad/audio-features-and-lyrics-of-spotify-songs. [Accessed: 16-Nov-2022].

[6] Huggingface, "Transformers/examples/research_projects/distillation at Main · Huggingface/Transformers," *GitHub*. [Online]. Available: https://github.com/huggingface/transformers/tree/main/examples/research_projects/distillation#how-to-use-distilbert. [Accessed: 21-Nov-2022].

[4] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," *Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019*, Mar. 2020. Available: https://doi.org/10.48550/arXiv.1910.01108.

[5] A. Boonyanit and A. Dahl, "Music Genre Classification using Song Lyrics," 2021.